gaussianKernel.m

```
function sim = gaussianKernel(x1, x2, sigma)
%RBFKERNEL returns a radial basis function kernel between x1 and x2
%   sim = gaussianKernel(x1, x2) returns a gaussian kernel between x1 and x2
%   and returns the value in sim

% Ensure that x1 and x2 are column vectors
x1 = x1(:); x2 = x2(:);

% You need to return the following variables correctly.
sim = 0;

sim = exp(-(sum((x1-x2).^2)/(2*sigma^2)));

end
```

dataset3Params.m

```
function [C, sigma] = dataset3Params(X, y, Xval, yval)
%DATASET3PARAMS returns your choice of C and sigma for Part 3 of the exercise
%where you select the optimal (C, sigma) learning parameters to use for SVM
%with RBF kernel

% You need to return the following variables correctly.
C = 1;
sigma = 0.3;

maxError = Inf;

% It would be nice if this didn't do function optimization by exhaustive search :-/
for currC = [0.01 0.03 0.1 0.3 1 3 10 30]
  for currSigma = [0.01 0.03 0.1 0.3 1 3 10 30]
    model = svmTrain(X, y, currC, @(x1, x2) gaussianKernel(x1, x2, currSigma));

    predictions = svmPredict(model, Xval);
    predictionError = mean(double(predictions ~= yval));

    if predictionError < maxError
      maxError = predictionError;
      C = currC;
      sigma = currSigma;
    end
  end
end

end
```

processEmail.m

```
function word_indices = processEmail(email_contents)
%Preprocesses the body of an email and returns a list of indices of the
%words contained in the email.

% Load Vocabulary
vocabList = getVocabList();

% Init return value
word_indices = [];
```

```
10
11  % ========================== Preprocess Email ============================
12
13  % Find the Headers ( \n\n and remove )
14  % Uncomment the following lines if you are working with raw emails with the
15  % full headers
16
17  % hdrstart = strfind(email_contents, ([char(10) char(10)]));
18  % email_contents = email_contents(hdrstart(1):end);
19
20  % Lower case
21  email_contents = lower(email_contents);
22
23  % Strip all HTML
24  % Looks for any expression that starts with < and ends with > and replace
25  % and does not have any < or > in the tag it with a space
26  email_contents = regexprep(email_contents, '<[^<>]+>', ' ');
27
28  % Handle Numbers
29  % Look for one or more characters between 0-9
30  email_contents = regexprep(email_contents, '[0-9]+', 'number');
31
32  % Handle URLS
33  % Look for strings starting with http:// or https://
34  email_contents = regexprep(email_contents, ...
35                             '(http|https)://[^\s]*', 'httpaddr');
36
37  % Handle Email Addresses
38  % Look for strings with @ in the middle
39  email_contents = regexprep(email_contents, '[^\s]+@[^\s]+', 'emailaddr');
40
41  % Handle $ sign
42  email_contents = regexprep(email_contents, '[$]+', 'dollar');
43
44
45  % ========================== Tokenize Email ============================
46
47  % Output the email to screen as well
48  fprintf('\n==== Processed Email ====\n\n');
49
50  % Process file
51  l = 0;
52
53  while ~isempty(email_contents)
54
55      % Tokenize and also get rid of any punctuation
56      [str, email_contents] = ...
57         strtok(email_contents, ...
58                [' @$/#.-:&*+=[]?!(){},''">_<;%' char(10) char(13)]);
59
60      % Remove any non alphanumeric characters
61      str = regexprep(str, '[^a-zA-Z0-9]', '');
62
63      % Stem the word
64      % (the porterStemmer sometimes has issues, so we use a try catch block)
65      try str = porterStemmer(strtrim(str));
66      catch str = ''; continue;
```

```
    end;

    % Skip the word if it is too short
    if length(str) < 1
       continue;
    end

    % Look up the word in the dictionary and add to word_indices if
    % found
    % ====================================================================
    for i = 1:length(vocabList)
      if(strcmp(str, vocabList{i}))
        word_indices = [word_indices ; i];
      end
    end
    % ====================================================================

    % Print to screen, ensuring that the output lines are not too long
    if (l + length(str) + 1) > 78
        fprintf('\n');
        l = 0;
    end
    fprintf('%s ', str);
    l = l + length(str) + 1;

end

% Print footer
fprintf('\n\n=========================\n');

end
```

emailFeatures.m

```
function x = emailFeatures(word_indices)
%Takes in a word_indices vector and
%produces a feature vector from the word indices.

% Total number of words in the dictionary
n = 1899;

% You need to return the following variables correctly.
x = zeros(n, 1);

for i = 1:length(word_indices)
  x(word_indices(i))=1;
end

end
```